KNN

In [50]:
```
clf = neighbors.KNeighborsClassifier(n_neighbors = 10, weights = 'distance', metric='euclidean')
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print("KNN accuracy: ", accuracy_score(y_test, y_pred))
print("KNN f1 score: ", f1_score(y_test, y_pred, average='weighted'))
print("KNN confusion matrix: ")
print(confusion_matrix(y_test, y_pred))
```

```
KNN accuracy:  0.8518518518518519
KNN f1 score:  0.8546870914020672
KNN confusion matrix:
[[11  1  0]
 [ 0  7  0]
 [ 0  3  5]]
```

In [51]:

In [52]:
```
df3['predicted_variety'] = y_pred
df3
```

```
C:\Users\Dell service\AppData\Local\Temp\ipykernel_3060\2710887449.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df3['predicted_variety'] = y_pred
```

Out[52]:

	sepal.length	sepal.width	petal.length	petal.width	variety	predicted_variety
9	4.9	3.1	1.5	0.1	NaN	Setosa
11	4.8	3.0	1.6	0.2	NaN	Setosa
17	5.1	3.5	1.4	0.3	NaN	Setosa
27	5.2	3.5	1.5	0.2	NaN	Setosa
33	5.5	4.2	1.4	0.2	NaN	Setosa
50	7.0	3.2	4.7	1.4	NaN	Versicolor
68	6.2	2.2	4.5	1.5	NaN	Versicolor
70	5.9	3.2	4.8	1.8	NaN	Virginica
71	6.1	2.8	4.0	1.3	NaN	Versicolor
81	5.5	2.4	3.7	1.0	NaN	Versicolor
91	6.1	3.0	4.6	1.4	NaN	Versicolor
115	6.4	3.2	5.3	2.3	NaN	Virginica
118	7.7	2.6	6.9	2.3	NaN	Virginica
122	7.7	2.8	6.7	2.0	NaN	Virginica
144	6.7	3.0	5.7	2.5	NaN	Virginica

Logistic Regression

In [53]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
```

```
regressor = LogisticRegression()
regressor.fit(X_train, y_train)

# predict
predictions = regressor.predict(X_test)
# evaluate
print("LR classification accuracy", accuracy_score(y_test, predictions))

# matrix confusion
cm = confusion_matrix(y_test, predictions)
print(cm)

# f1 score
print("LR f1 score", f1_score(y_test, predictions, average="macro"))
```

```
LR classification accuracy 0.8888888888888888
[[12  0  0]
 [ 0  7  0]
 [ 0  3  5]]
LR f1 score 0.8642533936651584
```

In [57]:
```
df4['predicted_variety'] = predictions
df4
```

Out[57]:

	sepal.length	sepal.width	petal.length	petal.width	variety	predicted_variety
9	4.9	3.1	1.5	0.1	NaN	Setosa
11	4.8	3.0	1.6	0.2	NaN	Setosa
17	5.1	3.5	1.4	0.3	NaN	Setosa
27	5.2	3.5	1.5	0.2	NaN	Setosa
33	5.5	4.2	1.4	0.2	NaN	Setosa
50	7.0	3.2	4.7	1.4	NaN	Versicolor
68	6.2	2.2	4.5	1.5	NaN	Versicolor
70	5.9	3.2	4.8	1.8	NaN	Virginica
71	6.1	2.8	4.0	1.3	NaN	Versicolor
81	5.5	2.4	3.7	1.0	NaN	Versicolor
91	6.1	3.0	4.6	1.4	NaN	Versicolor
115	6.4	3.2	5.3	2.3	NaN	Virginica
118	7.7	2.6	6.9	2.3	NaN	Virginica
122	7.7	2.8	6.7	2.0	NaN	Virginica
144	6.7	3.0	5.7	2.5	NaN	Virginica

SVM

In [58]:
```
# svm to classify
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score


clf = SVC(kernel='linear')
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print("SVM accuracy: ", accuracy_score(y_test, y_pred))
print("SVM f1 score: ", f1_score(y_test, y_pred, average='weighted'))
print("SVM confusion matrix: ")
```

In [58]:
```
# svm to classify
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score


clf = SVC(kernel='linear')
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print("SVM accuracy: ", accuracy_score(y_test, y_pred))
print("SVM f1 score: ", f1_score(y_test, y_pred, average='weighted'))
print("SVM confusion matrix: ")
print(confusion_matrix(y_test, y_pred))
```

```
SVM accuracy:  0.8518518518518519
SVM f1 score:  0.8436213991769547
SVM confusion matrix:
[[12  0  0]
 [ 0  7  0]
 [ 0  4  4]]
```

In [62]:
```
df5['predicted_variety'] = predictions
df5
```

Out[62]:

	sepal.length	sepal.width	petal.length	petal.width	variety	predicted_variety
9	4.9	3.1	1.5	0.1	NaN	Setosa
11	4.8	3.0	1.6	0.2	NaN	Setosa
17	5.1	3.5	1.4	0.3	NaN	Setosa
27	5.2	3.5	1.5	0.2	NaN	Setosa
33	5.5	4.2	1.4	0.2	NaN	Setosa
50	7.0	3.2	4.7	1.4	NaN	Versicolor
68	6.2	2.2	4.5	1.5	NaN	Versicolor
70	5.9	3.2	4.8	1.8	NaN	Virginica
71	6.1	2.8	4.0	1.3	NaN	Versicolor
81	5.5	2.4	3.7	1.0	NaN	Versicolor
91	6.1	3.0	4.6	1.4	NaN	Versicolor
115	6.4	3.2	5.3	2.3	NaN	Virginica
118	7.7	2.6	6.9	2.3	NaN	Virginica
122	7.7	2.8	6.7	2.0	NaN	Virginica
144	6.7	3.0	5.7	2.5	NaN	Virginica

turning hyperparameter for svm

In [65]:
```
# print best parameter after tuning
print(grid.best_params_)
# print how our model looks after hyper-parameter tuning
print(grid.best_estimator_)
# print accuracy score
print("SVM accuracy: ", accuracy_score(y_test, grid.predict(X_test)))
print("SVM f1 score: ", f1_score(y_test, grid.predict(X_test), average='weighted'))
print("SVM confusion matrix: ")
print(confusion_matrix(y_test, grid.predict(X_test)))
```

```
{'C': 1, 'gamma': 1, 'kernel': 'rbf'}
SVC(C=1, gamma=1)
SVM accuracy:  0.9629629629629629
SVM f1 score:  0.9629629629629629
SVM confusion matrix:
[[12  0  0]
 [ 0  7  0]
 [ 0  1  7]]
```

Out[67]:

	sepal.length	sepal.width	petal.length	petal.width	variety	predicted_variety
9	4.9	3.1	1.5	0.1	NaN	Setosa
11	4.8	3.0	1.6	0.2	NaN	Setosa
17	5.1	3.5	1.4	0.3	NaN	Setosa
27	5.2	3.5	1.5	0.2	NaN	Setosa
33	5.5	4.2	1.4	0.2	NaN	Setosa
50	7.0	3.2	4.7	1.4	NaN	Versicolor
68	6.2	2.2	4.5	1.5	NaN	Versicolor
70	5.9	3.2	4.8	1.8	NaN	Virginica
71	6.1	2.8	4.0	1.3	NaN	Versicolor
81	5.5	2.4	3.7	1.0	NaN	Versicolor
91	6.1	3.0	4.6	1.4	NaN	Versicolor
115	6.4	3.2	5.3	2.3	NaN	Virginica
118	7.7	2.6	6.9	2.3	NaN	Virginica
122	7.7	2.8	6.7	2.0	NaN	Virginica
144	6.7	3.0	5.7	2.5	NaN	Virginica

Random forests

In [69]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
```

In [70]:
```
clf = RandomForestClassifier(n_estimators=100, random_state=0)

# train model
clf.fit(X_train, y_train)

# predict
y_pred = clf.predict(X_test)

# evaluate
print("RF accuracy: ", accuracy_score(y_test, y_pred))
print("RF f1 score: ", f1_score(y_test, y_pred, average='weighted'))
print("RF confusion matrix: ")
print(confusion_matrix(y_test, y_pred))
```

```
RF accuracy:  1.0
RF f1 score:  1.0
RF confusion matrix:
[[12  0  0]
 [ 0  7  0]
 [ 0  0  8]]
```

```
df8
```

Out[74]:

	sepal.length	sepal.width	petal.length	petal.width	variety	predicted_variety
9	4.9	3.1	1.5	0.1	NaN	Setosa
11	4.8	3.0	1.6	0.2	NaN	Setosa
17	5.1	3.5	1.4	0.3	NaN	Setosa
27	5.2	3.5	1.5	0.2	NaN	Setosa
33	5.5	4.2	1.4	0.2	NaN	Setosa
50	7.0	3.2	4.7	1.4	NaN	Versicolor
68	6.2	2.2	4.5	1.5	NaN	Versicolor
70	5.9	3.2	4.8	1.8	NaN	Virginica
71	6.1	2.8	4.0	1.3	NaN	Versicolor
81	5.5	2.4	3.7	1.0	NaN	Versicolor
91	6.1	3.0	4.6	1.4	NaN	Versicolor
115	6.4	3.2	5.3	2.3	NaN	Virginica
118	7.7	2.6	6.9	2.3	NaN	Virginica
122	7.7	2.8	6.7	2.0	NaN	Virginica
144	6.7	3.0	5.7	2.5	NaN	Virginica

In [ ]: